An Efficient Method for Fully Relativistic Simulations of Coalescing Binary Neutron Stars

Walter Landry
Physics Dept., University of Utah, SLC, UT 84112
and Center for Radiophysics and Space Research, Cornell University, Ithaca, NY 14853

Saul A. Teukolsky
Center for Radiophysics and Space Research, Cornell University, Ithaca, NY 14853

The merger of two neutron stars has been proposed as a source of gamma-ray bursts, r-process elements, and detectable gravitational waves. Extracting information from observations of these phenomena requires fully relativistic simulations. Unfortunately, the only demonstrated method for stably evolving neutron stars requires solving elliptic equations at each time step, adding substantially to the computational resources required. In this paper we present a simpler, more efficient method. The key insight is in how we apply numerical diffusion. We perform a number of tests to validate the method and our implementation. We also carry out a very rough simulation of coalescence and extraction of the gravitational waves to show that the method is viable if realistic initial data are provided.

PACS number(s): 97.80.-d, 04.25.Dm, 04.40.Dg, 97.60.Jd

I. INTRODUCTION

After the discovery of the first binary neutron star system 1913+16, it was quickly realized that the orbit was decaying and the two stars would collide within a Hubble time. Because of the enormous available gravitational binding energy, neutron star-neutron star mergers like this became a popular mechanism for γ-ray bursts (e.g., [1]). The merger could also eject some neutron star material, which led some to propose that the subsequent rapid decompression of the nuclear density material could create r-process elements [2–4].

However, whether or not neutron star - neutron star mergers can explain these phenomena is still uncertain. What is reasonably certain is that, if general relativity is correct, the coalescence will emit a gravitational wave signal visible at cosmological distances to detectors being built or planned today [5]. By comparing the signals at coalescence to theoretical predictions, we may be able to learn more about the neutron stars themselves: their mass, radius, and internal structure. Because neutron stars are such dense, relativistic objects, such a comparison could also lead to new understandings in the physics at nuclear densities as well as the first strong field test of general relativity.

When the stars collide, shocks form, black holes appear, and in general the physics is very complicated. This requires a fully dynamical scheme to evolve the gravitational and matter fields. There have been a number of Newtonian and post-Newtonian simulations (see [6] and references therein). This approach expands the gravitational field equations around the Newtonian limit, using the ratio of the mass to radius, M/R, as a small parameter (We use units where $G = c = 1$). The problem is that the coalescence is highly relativistic. The quantity $2M/R = 0.4$ for the canonical neutron star with $M = 1.4M_\odot$, $R = 10$km. It goes up to $2M/R = 1$ when the black hole forms, a process that Newtonian and post-Newtonian simulations cannot even capture.

To treat this case, there have been some attempts to do a fully relativistic simulation, but with limited success. General relativity is a very complicated theory. Adding in the details of a numerical implementation leads to a multitude of things that can go wrong. As a result, self-consistent hydrodynamics+relativity simulations have had their greatest successes in only one or two dimensions [7–12]. These calculations have to evolve both the hydrodynamics (the matter) and the gravitational fields (the metric). In addition, general relativity allows the freedom of choosing coordinates. The calculations have all been carried out with variations of the formalism developed by Arnowitt, Deser and Misner (ADM) [13]. These lower dimensional calculations used a few tricks and applied brute force to manage long evolutions. In three dimensions, these techniques are either unavailable or impractical.

The neutron star problem differs from evolving black holes alone. In one sense, black holes are easier because there are no difficulties associated with the hydrodynamics or the (physically) sharp matter distribution at the surface. On the other hand, event horizons and singularities pose significant problems for numerical implementations in the black hole case, which have yet to be solved in three dimensions.

Undaunted, Nakamura *et. al.* [6] made some pioneering calculations of coalescing neutron stars. They came up against the same problem that has plagued numerical relativity until recently: growing instabilities that quickly crash the code. Marronetti *et. al.* [14] circumvented this problem by using a simplified metric. Unfortunately, using the simplified metric precludes an accurate determination of the gravitational waveform.

Font *et. al.* [15] have constructed a 3D code that has made some short calculations [16]. They have managed long *hydrodynamic* evolutions of single neutron stars (i.e., they kept the gravitational fields fixed to their analytic values), but a long term, self-consistent, stable evolution remained elusive.

Recently, Baumgarte *et. al.* [17] modified the original ADM equations, trying to remove pathologies from the equations which might lead to instabilities. The trade-off is that the implementation is a little more complex. They applied this formalism to a number of spacetimes [18], including a static neutron star. In contrast to the work in [15], they evolved the gravitational fields, but kept the hydrodynamic variables frozen to the analytic solution. They were rewarded with long term stability for this pseudo-dynamic problem.

Baumgarte *et. al.* [19] applied this method to a fully self-consistent hydrodynamic evolution of a neutron star. They were able to evolve the star for several dynamical times, but the numerical errors eventually caused the star to collapse upon itself. It is also not clear whether their choice of coordinates will work with two coalescing stars. The motion of two stars around the center of mass could drag the coordinates along. This twisting around the center could lead to coordinate singularities, ruining an otherwise physically valid simulation.

To alleviate this twisting, Shibata [20] adopted a more robust coordinate condition (technically, approximate minimal distortion), and got a long term, stable, self-consistent, 3D relativistic simulation of coalescence. Unfortunately, the coordinate condition is computationally expensive (it requires solving elliptic equations at every time step), and will significantly slow down any simulation.

This paper presents a simpler, more efficient method than [20] for evolving relativistic stars. We take elements from many different previous investigations, add in a few small new contributions, and combine them into a stable code. We use the original ADM formalism to evolve the metric. Although numerical relativists seem to shun them, we choose fully harmonic coordinates because they may help alleviate coordinate pathologies. There have been some attempts to use harmonic *slicing* (e.g., [21]), but not the full harmonic gauge. We use the relatively new and sophisticated high resolution shock capturing method for relativistic hydrodynamics developed in [15]. For the outer boundaries, we adopt the condition from [17] that allows gravitational waves to propagate off the grid. Our new contribution is how we add in a small numerical diffusion to stabilize the simulation. The usual way to add it in diffuses mostly along coordinate axes (x, y, z). We use a new scheme that also diffuses along diagonals ($x = y$, $x = -y$, $x = z$, $x = -z$, $y = z$, $y = -z$). This small variation allows long term, accurate evolutions.

We then present a number of tests: short term tests to validate our implementation of the equations, and long term tests to validate the whole approach. Then we proceed to compute a very unrealistic simulation of coalescing binary neutron stars to demonstrate that the method will work. Finally, we describe what remains to be done to turn this into a realistic simulation. We expect that reasonably accurate simulations will become available in the next few years, although they may require the largest supercomputers.

II. METHODS

A. Metric Evolution

To evolve the metric, we use the standard decomposition of Arnowitt, Deser and Misner (ADM) [13]. The line element then takes the form

$$ds^2 = g_{\mu\nu}dx^\mu dx^\nu = -\alpha^2 dt^2 + \gamma_{ij}(dx^i + \beta^i dt)(dx^j + \beta^j dt). \tag{1}$$

The lapse and shift functions α and β_i embody the gauge freedom of general relativity (i.e., the freedom to choose arbitrary coordinates), and so can be chosen without restriction. The embedding of the t = constant slices in the spacetime is described geometrically by the extrinsic curvature K_{ij}, which is defined by the equation

$$\frac{\partial \gamma_{ij}}{\partial t} = -2\alpha K_{ij} + \nabla_i \beta_j + \nabla_j \beta_i, \tag{2}$$

where ∇_i denotes a covariant derivative with respect to the three dimensional metric γ_{ij}. We have written this equation in the form of an evolution equation for γ_{ij}. Using Einstein's equations one can derive the evolution equation for K_{ij}:

$$\begin{aligned}\frac{\partial K_{ij}}{\partial t} &= -\nabla_i\nabla_j\alpha + K_{lj}\nabla_i\beta^l + K_{il}\nabla_j\beta^l + \beta^l\nabla_l K_{ij} \\ &+\alpha\left[R_{ij} - 2K_{il}K^l_j + KK_{ij} - S_{ij} - \frac{1}{2}\gamma_{ij}(\rho - S)\right].\end{aligned} \tag{3}$$

This in turn introduces the Ricci tensor R_{ij} and the matter terms ρ and S. R_{ij} is the three-dimensional Ricci tensor associated with the metric γ_{ij}. The most common way to write it involves first constructing the connection coefficients Γ^i_{jk} with the equation

$$\Gamma^i_{jk} = \frac{\gamma^{il}}{2}(\gamma_{lj,k} + \gamma_{lk,j} + \gamma_{jk,l}),$$

where commas denote partial derivatives. Then we could use the standard textbook formula to construct R_{ij}. Unfortunately, this involves taking derivatives of Γ^i_{jk}. In taking a derivative of a derivative numerically, there is no simple way to keep the error in the derivatives second order convergent and have the finite difference stencil include only nearest neighbor couplings. Second order accuracy requires knowledge of points two grid points away. This complicates boundary conditions, because we would have to apply them to a layer of points two points deep instead of one point deep. So, we write the Ricci tensor in an alternative form as

$$R_{ij} = \frac{1}{2}\gamma^{kl}\left[\gamma_{kj,il} + \gamma_{il,kj} - \gamma_{kl,ij} - \gamma_{ij,kl} + 2\left(\Gamma^m_{il}\Gamma_{mkj} - \Gamma^m_{ij}\Gamma_{mkl}\right)\right]. \tag{4}$$

Now, we can take centered, second-order differences of the metric γ_{ij}, for which we only need to use the nearest neighbors.

ρ and S are projections of the four-dimensional stress energy tensor $T^{\mu\nu}$ onto the three-dimensional spacetime described by γ_{ij}. They are defined in terms of the 4-vector normal to the three dimensional slice

$$n_\mu = (-\alpha, 0, 0, 0).$$

We then define three projections, ρ, J^i, and S_{ij} as

$$\begin{aligned} \rho &= 8\pi n_\mu n_\nu T^{\mu\nu} = 8\pi\alpha^2 T^{tt}, \\ J^i &= -8\pi n_\mu \gamma^i_j T^{\mu j}, \\ S_{ij} &= 8\pi\gamma_{ik}\gamma_{jl}T^{kl}, \end{aligned}$$

and

$$S = \gamma^{ij}S_{ij}.$$

B. Constraints

The evolution equations (eqs. 2 and 3) involve only 6 of the 10 Einstein equations. The remaining equations are the energy, or Hamiltonian, constraint

$$R + K^2 - K_{ij}K^{ij} = 2\rho, \tag{5}$$

and the three momentum constraints

$$\nabla_j\left(K^{ij} - \gamma^{ij}K\right) = J^i. \tag{6}$$

If we start with data that satisfies these constraints and evolve them with eqs. 2 and 3, then we are guaranteed to evolve to a spacetime that still satisfies the constraints. (Note that it doesn't matter how α and β^i are evolved. Their evolution is determined by the gauge, and eqs. 5 and 6 are true in any gauge.)

This guarantee is analytic, but errors in a numerical evolution can lead to a (γ_{ij}, K_{ij}) pair that does not exactly satisfy the constraints. Accordingly, we generally use the constraints as a check on the accuracy of the code. However, in order to get self-consistent initial data for two neutron stars, we need to explicitly enforce the constraints. In doing this, an ambiguity immediately presents itself. Eqs. 5 and 6 do not declare what combination of the twelve components of γ_{ij} and K_{ij} should be constrained by the four constraints. A natural way, but by no means the only way, is to decompose γ_{ij} and K_{ij} with the York decomposition [22]. We decompose the metric by defining a conformal metric $\widetilde{\gamma}_{ij} = \gamma_{ij}\psi^{-4}$, where $\det\widetilde{\gamma}_{ij} = 1$. The inverse of $\widetilde{\gamma}_{ij}$ is then $\widetilde{\gamma}^{ij} = \gamma^{ij}\psi^4$. We then apply the energy constraint to ψ.

Decomposing the extrinsic curvature is a little more complicated. We decompose the extrinsic curvature into traced and trace-free parts

$$K^{ij} = \psi^{-10}\left(\widetilde{A}^{ij} + (lX)^{ij}\right) + \frac{1}{3}\psi^{-4}\widetilde{\gamma}^{ik}\mathrm{TrK},$$

where

$$(lX)^{ij} = \widetilde{\nabla}^i X^j + \widetilde{\nabla}^j X^i - \frac{2}{3}\widetilde{\gamma}^{ij}\widetilde{\nabla}_k X^k,$$

and all covariant derivatives $\widetilde{\nabla}$ are with respect to the conformal metric $\widetilde{\gamma_{ij}}$. This is not a unique decomposition, since $\widetilde{A}^{ij}$ and $(lX)^{ij}$ describe overlapping degrees of freedom. In practice, we decompose K_{ij} assuming $X^i = 0$, solve for a new X^i which makes K_{ij} solve the constraints, and recompose K_{ij}.

In terms of these variables, the constraints become

$$\begin{aligned} -8\widetilde{\nabla}^2\psi &= -\widetilde{R}\psi - \frac{2}{3}(trK)^2\psi^5 \\ &\quad + \left(\widetilde{A}^{ij} + (lX)^{ij}\right)^2\psi^{-7} + 2\rho\psi^5, \end{aligned} \tag{7}$$

$$\begin{aligned} &\widetilde{\nabla}^2 X^i + \frac{1}{3}\widetilde{\nabla}^i\widetilde{\nabla}_j X^j + \widetilde{R}^i_j X^j \\ &\quad = J^i\psi^{10} - \widetilde{\nabla}_j\widetilde{A}^{ij} + \frac{2}{3}\psi^6\widetilde{\nabla}^i trK, \end{aligned} \tag{8}$$

where $\widetilde{R}$ is the Ricci tensor associated with the conformal metric $\widetilde{\gamma}$. To solve these equations, we first linearize them, writing ψ as $\psi_0 + \delta\psi$ and X^i as $X^i_0 + \delta X^i$. We explicitly decouple the linearized equations by setting $\delta X^i = 0$ in the linearized form of eq. 7, and $\delta\psi = 0$ in the linearized form of eq. 8. The linearized equations then become

$$\begin{aligned} -8\widetilde{\nabla}^2(\psi_0 + \delta\psi) &= -\widetilde{R}(\psi_0 + \delta\psi) - \frac{2}{3}(trK)^2\psi_0^4(\psi_0 + 5\delta\psi) \\ &\quad + \left(\widetilde{A}^{ij} + (lX_0)^{ij}\right)^2\psi_0^{-8}(\psi_0 - 7\delta\psi) + 2\rho\psi_0^4(\psi_0 + 5\delta\psi), \end{aligned} \tag{9}$$

$$\begin{aligned} &\nabla^2(X^i_0 + \delta X^i) + \frac{1}{3}\widetilde{\nabla}^i\widetilde{\nabla}_j(X^j_0 + \delta X) + \widetilde{R}^i_j(X^j_0 + \delta X^i) \\ &\quad = J^i\psi_0^{10} - \widetilde{\nabla}_j\widetilde{A}^{ij} + \frac{2}{3}\psi_0^6\widetilde{\nabla}^i trK. \end{aligned} \tag{10}$$

Then we solve the linearized equations using multigrid methods [23]. We use red-black Gauss-Seidel smoothing, with 2 pre- and post-smoothings for the linearized ψ equation, and 5 pre- and post-smoothings for the linearized X^i equation. On the outer boundaries, we assume that $\psi = \psi_{\mathrm{original}}$ and $X^i = X^i_{\mathrm{original}} = 0$ (i.e., we freeze the boundaries). When working on parallel machines, we smooth the solution on each machine individually, and then synchronize the edges. Then we add

$\delta\psi$ to ψ_0 and δX^i to X^i_0, recompute the terms involving ψ_0 and X^i_0 in eqs. 9 and 10, and iterate until we reach a tolerance of 10^{-10} in the norm of eqs. 9 and 10.

The explicit decoupling helps the elliptic solver converge. Decoupling does not affect the results, since we recompute the non-linear parts of eqs. 9 and 10 after each solution of the linearized equations.

As mentioned before, the constraints are only solved to get self-consistent initial data. The constraints are not solved during the evolution.

C. Coordinate Evolution

In general relativity you have the freedom to choose the coordinates however you wish. This freedom manifests itself in the choice of original coordinates as well as the evolution equations for the lapse α and the shift β^i. It is very easy to make a bad choice of coordinates. For example, consider a small perturbation in the original coordinates x^μ

$$x^\mu \rightarrow x^\mu + \xi^\mu. \tag{11}$$

A seemingly benign choice of evolution equations for α and β^i can make the small perturbation grow exponentially, creating coordinate singularities in an otherwise non-singular evolution. To prevent any bad behavior in ξ^μ, we can use harmonic coordinates. These coordinates satisfy

$$\Box x^\mu = 0. \tag{12}$$

That is, the coordinates obey a wave equation (although with covariant derivatives). Since eq. 12 is linear, and the original coordinates x^μ obey eq. 12, we then get the condition on ξ^μ

$$\Box \xi^\mu = 0. \tag{13}$$

Thus, ξ^μ *should* be wave like, and not exponentially increasing.

Imposing eq. 12 implies evolution equations for the lapse and shift. In practice, we use the evolution equation for $g_{tt} = -\alpha^2 + \beta^i\beta_i$ instead of the lapse. Then the equations are

$$\frac{\partial g_{tt}}{\partial t} = \left(\gamma^{ij}\alpha^2 - \beta^i\beta^j\right)\left(-\gamma_{ij,t} + 2\beta_{i,j}\right) + 2\beta^i g_{tt,i} \tag{14}$$

and

$$\begin{aligned}\frac{\partial \beta_k}{\partial t} &= 2\beta^i\left(\gamma_{ki,t} - \beta_{i,k} + \beta_{k,i}\right)\\ &\quad - \left(\gamma^{ij}\alpha^2 - \beta^i\beta^j\right)\left(\gamma_{ij,k} - 2\gamma_{kj,i}\right) + g_{tt,k}.\end{aligned} \tag{15}$$

D. Matter Evolution

Our method is exactly the same as in [15] except for our treatment of low density regions and our time stepping. We briefly summarize the method here.

We represent the matter with three variables — D, τ , and S_i — which roughly correspond to the mass density, energy density, and momentum density. We can then write the evolution equations in the form

$$\frac{\partial(\text{variable})}{\partial t} + \partial_i(\text{flux})^{\text{i}} = (\text{source}),$$

where the only spatial derivatives of the *matter* variables occur in the $\partial_i(\text{flux})^{\text{i}}$ term. Writing the equations in this form allows us to apply the cornucopia of shock capturing methods developed over the years for ordinary hydrodynamics. We use a Roe scheme with a standard min-mod piecewise-linear reconstruction algorithm [24].

The fluxes and sources are written in terms of primitive variables $\{\rho, \epsilon, v_i\}$ which are not simply related to the evolved variables $\{D, \tau, S_i\}$. Thus we have to do a costly root finding at each point at each time level to find $\{\rho, \epsilon, v_i\}$ from $\{D, \tau, S_i\}$. This step can go awry in low density regions, because numerical errors can conspire to evolve $\{D, \tau, S_i\}$ into something that has no physical $\{\rho, \epsilon, v_i\}$. To combat this, we use a variation of the method of [15]. We create a fake "atmosphere" in the empty space around the star by setting a minimum value of D of about 10^{-9} of the initial central density of the star. Whenever D is evolved to a value lower than 10^{-9}, we set D to 10^{-9}. Also, if D is lower than 10^{-5}, we set τ and S_i to zero. Furthermore, if we ever get a transform from $\{D, \tau, S_i\}$ to $\{\rho, \epsilon, v_i\}$ that gives unphysical values (e.g., negative energies), we start over, replacing the definition of τ with the condition for adiabatic flow $P = k\rho^\Gamma$. We then compute $\{\rho, \epsilon, v_i\}$ and recompute τ. This simple prescription allows long term evolutions of neutron stars and their surroundings.

Apart from the low density treatment, the only other difference between us and [15] is the time stepping. We implement a Strang split of the hydrodynamics [24]. That is, we evolve for a half time step as if the evolution equations are only the flux terms of the hydrodynamics. Then we evolve for a full time time step as if the evolution equations are only the hydrodynamic source terms and all the metric and coordinate terms. Then we again evolve for a half time step as if the evolution equations are only the flux terms.

When we evolve the fluxes for the first half step, we evolve first in one randomly chosen direction d_1, then in another randomly chosen direction d_2, and then in the third direction d_3. For the second half-step, we do it in the reverse order d_3-d_2-d_1. Thus, for each step we choose from one of six random orders: *xyz*-*zyx*, *xzy*-*yzx*, *yxz*-*zxy*, *yzx*-*xzy*, *zxy*-*yxz*, and *zyx*-*xyz*.

We use second-order Runge-Kutta stepping for the matter, metric, and coordinate equations. That is, when evolving the equation

$$\frac{\partial Q}{\partial t} = F(Q)$$

from t to $t + \Delta t$, we first take an intermediate step

$$Q_{\rm intermediate} = Q_t + \frac{\Delta t}{2} F(Q_t),$$

and then we take the full step

$$Q_{t+\Delta t} = Q_t + \Delta t\, F(Q_{\rm intermediate}).$$

In conjunction with the Strang split, this makes the whole evolution second-order accurate. We use a Courant factor $\Delta t / \Delta x = 0.25/c$. This scheme is different from [15]. They evolve the fluxes in the x, y, and z direction all at once, and their coupling in time between hydrodynamics and spacetime is very different.

E. Boundary Conditions

On the outer boundaries, we use the interpolated Sommerfeld condition of [17] for the metric and coordinate terms. Essentially, we assume that all of the metric and coordinate variables behave like outgoing, radial waves

$$Q(t,r) = \frac{G(\alpha t - (\det \gamma)^{\frac{1}{6}} r)}{r},$$

where $Q = \{\gamma_{ij} - \delta_{ij}, K_{ij}, g_{tt} + 1, \beta_i\}$. Thus, we compute the value at the boundary by following the characteristic back to the previous time step and linearly interpolating the corresponding variable to that point. We tried just freezing the boundaries (i.e., $\gamma_{\rm old} = \gamma_{\rm new}$), but that seemed to reflect and amplify waves, leading to an instability.

For the matter terms, we use "flat" boundaries. That is, values at points on the boundary are set equal to those at the points just inside the boundary. Although this condition seems crude, it works rather well at propagating blobs of matter off of the computational grid [25].

F. Numerical Diffusion

Evolutions using the methods described up till now have failed catastrophically because of short wavelength numerical instabilities that grow without bound. To control these instabilities, we add some numerical diffusion. We do this by adding a ∇^4 term to the right hand sides of eqs. 2, 3, 14, and 15. This technique, introduced in [26], has the effect of spreading out short wavelength, poorly resolved features, while leaving long wavelength, well resolved features alone. This is exactly what is required — we don't need an accurate evolution of short wavelength features since they are poorly represented on a grid with finite spacing. Thus, eq. 2 becomes

$$\frac{\partial \gamma_{ij}}{\partial t} = -2\alpha K_{ij} + \nabla_i \beta_j + \nabla_j \beta_i - q\,(\Delta x)^3\, \nabla^4 \gamma_{ij}, \quad (16)$$

where q is a small constant (we use $q = 0.09/R_*^3$, where R_* is the radius of the neutron star). The $(\Delta x)^3$ factor ensures that, as the resolution increases, the modified equation quickly converges to the original continuum equation. We implement the ∇^4 as $\nabla^2\left(\nabla^2\right)$, and assume that $\nabla^2 = 0$ on the boundaries. This is compatible with the $1/r$ falloff, since $\nabla^2\,(1/r) = 0$ on the boundaries. The usual way to implement ∇^2 is as centered second derivatives along the coordinate axes ($\nabla^2 = \partial^2/\partial x^2 + \partial^2/\partial y^2 + \partial^2/\partial z^2$). Thus, a plot of the finite difference stencil in the x-y plane looks like Fig. 1.

This gives diffusion that acts primarily along the coordinate directions x, y, and z. Unfortunately, eq. 4 has mixed derivatives of γ_{ij} (e.g., $\partial^2 \gamma_{ij}/\partial x \partial y$), so points couple along diagonal directions. To fix this, we define new, diagonal directions $u = x + y$, $v = x - y$, $w = z$, and take derivatives along these directions. Then the finite difference stencil looks like Fig. 2, which couples along the diagonals in the x-y plane. We repeat this for the x-z and y-z planes and average the three different representations of ∇^2. This gives us a stencil that couples along the axes and diagonals. Finally, we apply this "diagonal diffusion" to the *new* time level. That is, ordinarily, without the diffusion, the evolution equation is implemented as

$$\gamma_{\rm new} = \gamma_{\rm old} + \Delta t (RHS),$$

where RHS is the right hand side of eq. 2. We add diffusion by changing the update to

$$\gamma_{\rm new} = \gamma_{\rm old} + \Delta t \{RHS - q \Delta x^3 \nabla^4 [\gamma_{\rm old} + \Delta t\,(RHS)]\}.$$

This is equivalent to first order to eq. 16. We implement it this way because short wavelength instabilities can sometimes change sign at each time step. Applying numerical diffusion to the old time step as in eq. 16 can end up being always a step out of sync with the instabilities, adding when it should be subtracting. Implementing it in this way doesn't allow the other parts of the equation (RHS) to interfere with the diffusion.

Adding explicit diffusion does slightly alter the equations away from the original physical ones, although the change gets smaller with higher resolution. In that sense, you can just consider it a slight addition to the error, getting smaller with more resolution. However, the error is not random; it moves error in the direction of too much diffusion. In an attempt to minimize this error, we tried applying this diffusion only to the coordinate evolution equations (eqs. 14, 15), because the coordinate evolution does not affect physical quantities. Yet even when we explicitly enforced the constraints at each time step, short wavelength spikes grew and destroyed the simulation.

This diffusion is very effective. Ordinarily, second-order Runge-Kutta is unconditionally unstable for most problems. In this case, instabilities seem to be dominated by other factors. We have tried alternative stepping alorithms, such as iterative Crank-Nicholson with varying numbers of iterations, and, in every case, the evolution was unstable without the diffusion, and stable with it.

G. Computational Issues

The entire code is written in C++ and runs on the Cornell Theory Center IBM SP2 supercomputer. The code has also been ported to a Sun workstation, and is freely available from the authors. The runs with 9^3, 17^3, and 33^3 points used 1 processor, the 65^3 runs used 8 processors and the 129^3 runs used 64 processors. We used the Kelp libraries [27] to split up the grid among multiple processors and simplify the message passing between processors. The code scales from 1 processor to 64 processors with about 70% parallel efficiency. All of the runs (except for the convergence tests, which were very short) took 12-72 hours, taking 1000-1500 steps every 12 hours. The hydrodynamics used about 80% of the CPU time, the metric and coordinate evolution used about 15%, and everything else took up the remaining 5%.

III. TESTS

We performed two different types of tests: short-term tests to verify the correct implementation of the equations, and long-term tests to verify the stability and accuracy of the method.

A. Short-Term Tests

To test the hydrodynamic evolution and its ability to handle shocks, we ran shock tubes as in [15]. We set up a one dimensional shock tube with $\rho = 10$, $P = 13.3$, $v = 0$, on the left and $\rho = 1$, $P = 0.66 \cdot 10^{-6}$, $v = 0$ on the right. This problem has an analytic solution [28,29] which we can compare to our evolution. Figure 3 shows a comparison of the normalized values of the pressure, density, and velocity, and we can see that our method reproduces the analytic answer reasonably well.

To test the metric and coordinate evolution, we ran convergence tests with the exterior metric of a black hole in full harmonic gauge [30]. To test all of the evolution terms and their coupling to each other all at once, we ran convergence tests on static and boosted neutron stars. We performed our neutron star tests with a $\Gamma = \frac{5}{3}$, $K = 5.380 \cdot 10^9 \mathrm{cm}^4 \mathrm{g}^{-2/3} \mathrm{s}^{-2}$ polytrope with a central density $\rho_{\mathrm{central}} = 1 \cdot 10^{15} \mathrm{g\,cm}^{-3}$, resulting in a mass $M_* = 1.35 \cdot 10^{33}$g, a Schwarzschild radius $R = 12.7$km, and thus a compaction $M/R = 0.08$. The maximum central density for this equation of state is about $\rho_{\max} = 3.79 \cdot 10^{15} \mathrm{g\,cm}^{-3}$, corresponding to a maximum mass of $1.51 \cdot 10^{33}$g. Then we converted the variables into the harmonic gauge [30] which gives us a harmonic radius of $R_* = 11.7$km. We boosted the stars with a variety of velocities $[\vec{v} = (0.3c, 0.2c, 0.1c), (0.9c, 0, 0), (0, 0.5c, 0)]$ using the prescription in [15]. We did convergence tests over both a cube entirely within the star and a cube entirely outside the star. We offset the grid by $10^{-5} R_*$ in the x, y, and z directions to avoid having to deal with spherical coordinate pathologies when setting up the initial configuration.

When doing convergence tests, we first took one step on a 9^3 grid, two steps on a 17^3 grid, and four steps on a 33^3 grid. All three grids spanned the same space, so each step doubled the resolution. Then we compared the solution on the three grids against the analytic solution. Because our differencing and time stepping are both second order, the error should decrease as the square of the number of points (i.e., a factor of four with each step in resolution).

However, this is not a perfect argument because there are higher order errors contributing to the overall error. Fortunately, these errors decrease even faster than the second order error. Therefore, if we plot the 9^3 error, the 17^3 error multiplied by 4, and the 33^3 error multiplied by 16, then we should see the 17^3 error closer to the 33^3 error than the 9^3 error.

Unfortunately, even this procedure is not foolproof. Sometimes the error can go through zero, leading to cases where the 17^3 error is closer to the 9^3 error than the 33^3 error in isolated regions. Figure 4 shows a plot where this happens. Sometimes the higher order error is still large, leading to cases where the high resolution error decreases so quickly that the medium resolution error is no longer closer to it. As an added difficulty, our outer boundaries aren't very good, so we couldn't use the error near it. In practice, we had to throw away the two outermost points, leaving only $5^3 = 125$ points to compare. In addition, our solution of the initial data for the neutron star is imperfect, so the center point was often bad (For example, the only bad point for γ_{xy} is at the center).

Finally, to get proper convergence, we had to turn off the numerical diffusion terms, because they add unphysical, although small, diffusion, and also the min-mod slope limiter in the hydrodynamics, because it enforces first order convergence near maxima. Keeping these techniques allows longer evolutions while sacrificing strict second-order convergence.

Even with all of these caveats, checking convergence was a powerful tool for finding bugs. An error in how we implement an equation prevents the entire grid from converging. Thus, a plot like Fig. 5, which shows the worst converging variable for that particular set of convergence tests, was not worrisome. We can trace its defects back to a combination of what is wrong in Fig. 4 and problems at the center. When we can not do that, the cause is a bug. We found many bugs this way. For example, we accidentally implemented the $-2\alpha K_{il} K^l_j$ term in eq. 3 as $-2\alpha K_{il} K^j_l$, and the indexing error prevented convergence. This sensitivity gives us a reasonable degree of confidence that we implemented the correct equations, and implemented them without error.

B. Long-Term Tests

To test the ability of our code to handle long neutron star evolutions, we ran the code with a single, stationary star. The idea is to see how well it holds a static star in equilibrium. In this section and later we adopt the units $R_* = 11.7\text{km} \equiv 1$, $M_* = 1.35 \cdot 10^{33}\text{gm} \equiv 1$, and $c \equiv 1$. This implies $G = 0.08$, $0.039\text{ms} = 1$, and the hydrodynamic timescale $\rho_{\text{central}}^{-1/2} = 4.49$. We compute the norm of a variable by averaging its square over the grid and then taking a square root.

The first set of tests evolved just the metric and coordinate terms, keeping the matter terms fixed to their analytic values. Figure 6 plots the error in the energy constraint (eq. 5) vs. time for three cases. It quickly drops down and levels off as the simulation settles into a stable configuration. Ideally, we would like that configuration to be the analytic one, but finite resolution, numerical diffusion, and the imperfect outer boundary condition will change it. Figure 7 plots a cross section of the error for the three cases in the final state. The errors for the low resolution, close and far boundaries cases are nearly identical. The only visible difference is near $x = 2$, where the close boundary has a little more error because of the boundary. It is also clear that better resolution helps a lot, drastically decreasing the error. This is probably because the biggest contributor to the error here is the numerical diffusion terms. Their effect decreases as the *cube* of the grid spacing, which is much faster than the other errors, which decrease as the *square* of the grid spacing. The reason that the constraint norm for the far boundary case is so much smaller than the close boundary case is that the error is concentrated in the center near the star. The far boundary case just has more points over which to average the error. This is encouraging, because it means our simulation will improve as we increase resolution (not always a given), and our boundary is probably adequate, even relatively close in.

Figure 8 plots a cross section of the relative error in $\sqrt{\det \gamma_{ij}}$. Even for the worst case, the errors are only about 1%. The effect of the boundaries is much stronger than in Fig. 7. The simulation does not stray far from the analytic solution, giving us hope that numerical diffusion and the boundary conditions do not significantly affect the results.

The second set of tests evolved just the hydrodynamics, keeping the metric and coordinate terms fixed to their analytic initial values. Figures 9 and 10 show a cross section of T^μ_μ versus time for two resolutions. The star rapidly diffuses out until it starts to interact with the boundaries. Remember that the hydrodynamic terms do not have any numerical diffusion. This diffusion doesn't actually cause a problem until late times, when "fingers" appear, stretching from the star to the boundary (Fig. 11). These fingers quickly grow until they dominate the simulation. The simulation still looks well behaved, if inaccurate, for a long time before then (Fig. 12).

The third set of tests evolved everything: hydrodynamics, metric, and coordinates. The results are largely the same as the hydrodynamics-only results (Figs. 13, 14, and 15). We also ran a test with far boundaries, and, as expected, we were able to evolve for significantly longer times. Figure 16 shows a plot of the central density T^μ_μ vs. time for the three cases. Eventually, the star again developed "fingers", at which time the code became hopelessly inaccurate.

IV. COALESCENCE

To show that the method will work for its intended purpose, simulating coalescing stars, we simulated a coalescence with easily computed, although not astrophysical, initial data. We took the equilibrium solution for the neutron star, replicated it next to itself, and then boosted the stars with $v = 0.15c$ in opposite directions as shown in Fig. 17 (the Kepler frequency for two point particles is $v = 0.19c$). To get self-consistent gravitational initial data, we solved the constraints with this matter source.

This is not astrophysically interesting initial data by any means. The equation of state, and thus the size, mass, and internal structure of the stars, are all wrong. Even if they were right, setting up the variables by placing boosted solutions next to each other is definitely wrong. There is likely to be a large amount of initial wave content, which can seriously affect the dynamics. We discovered this to our chagrin when we tried starting the stars a little farther apart. The initial wave content pushed the stars away from each other!

In addition, the resolution of these tests is really no better than the long term tests in section III B, which, as we saw, were not very accurate. Even so, it serves as an important validation of the method.

To measure the gravitational waves, we adopt a scheme similar to that used in [31]. We can define the transverse traceless (TT) gauge wave amplitudes h_+ and $h_\times$ by projecting out components of the metric. Along the z axis, this becomes

$$h_+ = \frac{1}{2}\left(\gamma_{xx} - \gamma_{yy}\right), \tag{17}$$

$$h_\times = \gamma_{xy}. \tag{18}$$

A more exact method would be to use gauge-invariant Moncrief variables [32] and integrate over spherical harmonics [33]. Considering the accuracy of the underlying simulation, this would be overkill.

Figure 18 shows the gauge invariant trace T^μ_μ vs. time for a low resolution run with the boundaries very far out at (−8,8). The stars complete about two orbits before completely merging.

Figures 19 and 20 show the amplitude of h_+ and $h_\times$ for three resolutions. As we increase the resolution, the waveforms seem to converge, revealing finer and finer details. The atmosphere was treated a little differently in these runs, with the floor value of D set at $10^{-4}D_{\rm central}$ instead of $10^{-9}D_{\rm central}$. This causes the evolution to go bad earlier, around $t = 60$. The earlier waveforms are not significantly affected, though.

If we run some simulations with the boundaries farther out, the results are not as encouraging. Plotting two resolutions of far boundary runs with the close boundary, high resolution run (Figs. 21, 22), we see that the initial amplitude and arrival time of the $h_\times$ wave is different. It arrives later, whereas if it were arriving from the $z = 0$ plane, where the stars are, it should arrive at an earlier time. This can not be explained from differences in how the waves propagate from $r = 4R_*$ to $r = 8R_*$. Figure 24 shows wave amplitudes for the low resolution, far boundary run extracted at different radii. The $h_\times$ amplitudes match each other very well.

Looking at the h_+ wave, we see new structure around $t = 10-15$ (twin spikes) that is absent in the close boundary runs. This is probably just propagation effects, for if we plot the wave amplitudes extracted at different radii (Fig. 23), we see that they match the structure much more at $z = 4$.

These differences in $h_\times$ probably arise from differences in the initial data. Remember that we solve the constraints to get the initial data. We assume that the outer boundaries are correct, and change the interior metric variables to fit. Thus, the different simulation domains will have varying amounts of initial wave content. This also suggests, unfortunately, that the big, initial peaks in h_+ and $h_\times$ may not be real.

The stress energy tensor for weak waves in the TT gauge is [34]

$$T_{\mu\nu} = \frac{1}{2}\left\langle h_{ij,\mu}h_{ij,\nu}\right\rangle,$$

where the $\langle\ldots\rangle$ symbol means an average over several wavelengths. The total energy that passes through a thin shell at radius r in time Δt is $4\pi T_{00}r^2c\Delta t$. To estimate $h_{ij,0}$, note that h_+ and $h_\times$ go from $\sim 10^{-2}$ to 0 over a time of $\sim 10R_*$. We extract the waves at $4R_*$, so the luminosity is

$$L \sim 4\pi \cdot \frac{1}{2}\frac{\left(h_+^2 + h_\times^2\right)\left(4R_*\right)^2}{\left(10R_*\right)^2} = 2\cdot 10^{-4}.$$

If we compute an estimate of the gravitational wave luminosity from the quadrupole formula for point masses in circular orbits [35, Section 36.6],

$$L = \frac{32}{5}\frac{\mu^3M^2}{a^5} \sim 3\cdot 10^{-7},$$

where $\mu = m_1m_2/M$, $M = m_1 + m_2$, a = separation = $2\mathrm{R}_*$. The large differences in magnitude suggest that the waves may be generated by something other the stars, such as the interaction between the right and left sides of the grid (Fig. 17). The spacetimes are initially boosted in opposite directions, so there is a discontinuity at $x = 0$. This discontinuity is smoothed somewhat, but not removed, when we solve the constraints.

V. FUTURE DIRECTIONS

There are a few things that must be done before the code will produce interesting results. First, the code needs realistic initial data along with more realistic equations of state. A number of groups [36–38] have managed to construct reasonable initial data. Getting these results and converting them into a usable format for the dynamical code is, in principle, a straightforward task.

Second, the code needs better resolution. Simply increasing the number of points is impractical, since the code already taxes the capabilities of current supercomputers. Instead, the solution is probably adaptive mesh refinement (AMR) [39]. Massively parallel machines will *still* be required, but at least then the problem is possible. Unfortunately, AMR on massively parallel machines is still a new endeavor, so libraries for making it straightforward to change a normal, parallel code into parallel AMR code are still immature. AMR will also allow us to move the boundaries out much farther, relieving some headaches there.

These steps will allow the code to be applied to the most pressing reason for these simulations: providing accurate theoretical waveforms for the new generation of gravitational wave detectors coming on line. It will also give reasonably accurate estimates of the amount of material ejected as possible r-process elements. To understand γ-ray bursts, on the other hand, will require this and more. Implementing magneto-hydrodynamics will allow us to evaluate, for example, the likelihood of the DRACO model [40]. Adding neutrino generation and transport will account for an important energy source. Also, neutrino effects may, as in supernovae, affect the dynamics [41].

ACKNOWLEDGEMENTS

We wish to thank Mark Scheel, Greg Cook and Larry Kidder for many useful discussions. We would also like to thank Mark Miller for answering questions about his numerical relativity code, Scott Baden for his help with getting Kelp to work, Roger Lovelace for his help with hydrodynamics, and Thomas Baumgarte for his help with outer boundary conditions. This work was supported in part by NSF grants PHY 98-00737 and 99-00672, and NASA Grant NAG5-7264 to Cornell University.

[1] R. Narayan, B. Paczynski, and T. Piran, Astrophys. J. **395**, L83 (1992)

[2] E. M. D. Symbalisty and D. N. Schramm, Astrophys. Lett. **22**, 143 (1982)

[3] D. Eichler, M. Livio, T. Piran, and D. N. Schramm, Nature **340**, 126 (1989)

[4] S. Rosswog, M. Liebendorfer, F.-K. Thielemann, M. B. Davies, W. Benz, and T. Piran, Astr. and Astrophys. **341**, 499 (1999)

[5] A. Abramovici *et. al.*, Science, **256**, 325 (1992)

[6] T. Nakamura and K. Oohara, gr-qc/9812054

[7] R. F. Stark and T. Piran, Phys. Rev. Lett. **65**, 1876 (1985)

[8] C. Evans, in *Dynamical Spacetimes and Numerical Relativity*, edited by J. Centrella (Cambridge Univ. Press, Cambridge, England, 1986), pp. 3-39

[9] P. J. Schinder, S. A. Bludmann, and T. Piran, Phys. Rev. D **37**, 2722 (1988)

[10] A. Mezzacappa and R. A. Matzner, Astrophys. J. **343**, 853 (1989)

[11] S. L. Shapiro and S. A.Teukolsky, Astrophys. J. **235**, 199 (1980)

[12] A. M. Abrahams, G. B. Cook, S. L. Shapiro, and S. A. Teukolsky, Phys. Rev. D **49**, 5153 (1994)

[13] R. Arnowitt, S. Deser, and C. W. Misner, in *Gravitation, An Introduction to Current Research*, edited by L. Witten (Wiley, New York, 1962), for a more accessible description, see [35].

[14] P. Marronetti, G. J. Mathews, and J. R. Wilson, Phys. Rev. D **58**, 107503 (1998)

[15] J. A. Font, M. Miller, W. Suen, and M. Tobias, gr-qc/9811015

[16] M. Miller, W. Suen, and M. Tobias, gr-qc/9904041

[17] T. W. Baumgarte and S. L. Shapiro, Phys. Rev. D **59** 024007 (1999)

[18] T. W. Baumgarte, S. A. Hughes, and S. L. Shapiro, Phys. Rev. D **60**, 087501 (1999)

[19] T. W. Baumgarte, S. A. Hughes, L. Rezzolla, S. L. Shapiro, and M. Shibata, gr-qc/9907098

[20] M. Shibata, Phys. Rev. D **60**, 104052 (1999); M. Shibata and K. Uryu, gr-qc/9911058

[21] C. Bona, J. Masso, E. Seidel, and P. Walker, gr-qc/9804052

[22] J. W. York Jr. and T. Piran, in *Spacetime and Geometry: The Alfred Schild Lectures*, edited by R. Matzner and L. U. Shepley (Texas Press, Austin, 1982)

[23] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, (Cambridge University Press, Cambridge, England, 1992), pg. 871-888 or online at http://www.nr.com

[24] R. J. Leveque, Numerical Methods for Conservation Laws, 2nd Edition, (Birkhauser-Verlag, Basel, 1992)

[25] R. J. Leveque, 1997 J. Comput. Phys. **131**, 327 (1997)

[26] H. O. Kreiss and J. Oliger, *Methods for the Approximate Solution of Time Dependent Problems*, GARP Publication Series No. 10 (World Meteorological Organization, Geneva, 1973)

[27] S. B. Baden, http://www-cse.ucsd.edu/groups/hpcl/scg/kelp.html

[28] J. Centrella and J. R. Wilson, Astrophys. J. Supp. **54**, 229 (1984)

[29] V. Schneider, U. Katscher, D. H. Rischke, B. Waldhauser, and J. A. Maruhn, J. Comput. Phys. **105**, 92 (1993)

[30] G. B. Cook and M. A. Scheel, Phys. Rev. D **56**, 8 (1997)

[31] M. Shibata, Prog. Theor. Phys. **101**, 1199 (1999)

[32] V. Moncrief, Ann. Phys. **88**, 323 (1974)

[33] A. M. Abrahams and C. R. Evans, Phys. Rev. D **37**, 317 (1988); Phys. Rev. D **42**, 2585 (1990).

[34] S. L. Shapiro and S. A. Teukolsky, *Black Holes, White Dwarfs, and Neutron Stars: The Physics of Compact Objects* (Wiley-Interscience, New York, 1983)

[35] C. W. Misner, K. S. Thorne, and J. A. Wheeler, *Gravitation* (W. H. Freeman and Co., San Francisco, 1973)

[36] S. Bonazzola, E. Gourgoulhon, and J.-A. Marck, Phys. Rev. Lett. **82**, 892 (1999); gr-qc/9904040

[37] P. Marronetti, G. J. Mathews, and J. R. Wilson, Phys. Rev. D **60**, 087301 (1999); G. J. Mathews and J. R. Wilson, gr-qc/9911047

[38] K. Uryu and Y. Eriguchi, gr-qc/9908059

[39] M. S. Berger and P. J. Collela, J. Comput. Phys. **82**, 64 (1989)

[40] W. Kluzniak and M. Ruderman, Astrophys. J. **505**, L113 (1998)

[41] H. A. Bethe, Astrophys. J. **490**, 765 (1997)

FIG. 1. Finite difference stencil along axes. The circles denote which points ∇^2 samples in the x-y plane. The points with X's do not affect the value of ∇^2.

FIG. 2. Finite difference stencil along diagonals.

FIG. 3. Normalized values of the density ρ, pressure P, and velocity v for a one-dimensional shock tube at $t = 0.5$ using 256 grid points.

FIG. 4. Plots of the sign of the difference $\left|4 \cdot \left|17^3 \text{ error}\right| - \left|9^3 \text{ error}\right|\right| - \left|16 \cdot \left|33^3 \text{ error}\right| - 4 \cdot \left|17^3 \text{ error}\right|\right|$ for the variable D for a neutron star with the boost $\overrightarrow{v} = (0, 0.5c, 0)$. The boundaries are at $\pm 0.25 R_*$, but the outermost 2 points are discarded, so this grid is only 5 points on a side. When the difference is positive, the normalized 17^3 error is closer to the normalized 33^3 error than the 9^3 error, implying correct convergence. Thus, negative numbers (the dark areas) indicate bad convergence. Note that the plot only goes negative in limited regions. The 3rd order error goes through zero there, so the normalized errors cross each other.

FIG. 5. Plots as in Fig. 4 but for the variable K_{yz}. For this set of convergence tests, this is the variable that looks the worst.

FIG. 6. The norm of the error in eq. 5 vs. time when the matter terms are kept frozen for three cases: a low resolution run, a high resolution run, and another low resolution run with the boundaries twice as far out.

FIG. 7. The error in eq. 5 for the relaxed state on Fig. 6. The star is at the origin, and the star is spherically symmetric. This plot is taken from the $x > 0$, $y = z = 0$ line.

FIG. 8. A cross section of the relative error in $\sqrt{\det \gamma_{ij}}$ for the relaxed states of Fig. 6.

FIG. 9. A cross section of T^μ_μ for a low resolution run (33^3 points) when only the hydrodynamics are evolved. The star quickly spreads out, although the scale is very small.

FIG. 10. A cross section of T^μ_μ for a high resolution run (65^3 points) when only the hydrodynamics are evolved. The star spreads out much more slowly than the low resolution run of Fig. 9.

FIG. 11. A cross section of T^μ_μ in the equatorial plane taken at time $t = 68.5$. A finger has developed along the $x = 0$ line.

FIG. 12. A cross section of T^μ_μ in the equatorial plane taken at time $t = 50$. There is no evidence of a finger.

FIG. 13. A cross section of T^μ_μ for a low resolution run (33^3 points) when everything is evolved. Note that it is very similar to Fig. 9.

FIG. 14. A cross section of T^μ_μ for a high resolution run (65^3 points) when everything is evolved. Once again, very similar to the hydrodynamics-only result (Fig. 10).

FIG. 15. A cross section of T^μ_μ for a low resolution run, but with the boundaries twice as far as in Fig. 13. Note that it takes *much* longer to reach the boundaries, resulting in a much longer evolution.

FIG. 16. The central density of the star for three different runs. Note that the two low resolution cases track each other very well until the close boundary case diverges dramatically.

FIG. 17. Overhead view of the initial state of the two neutron stars

FIG. 18. T^μ_μ for coalescing stars. The simulation spans (-8,8), but we only plot (-2,2) to show the details of the merger.

FIG. 19. h_+ (eq. 17) for three different resolutions extracted at the boundary $z = 4$.

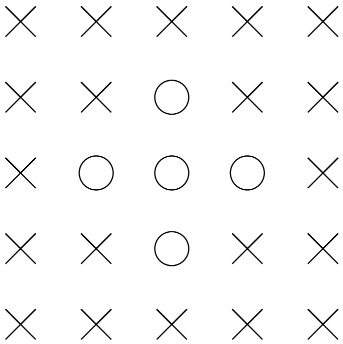
FIG. 20. $h_\times$ (eq. 18) for three different resolutions extracted at the boundary $z = 4$.

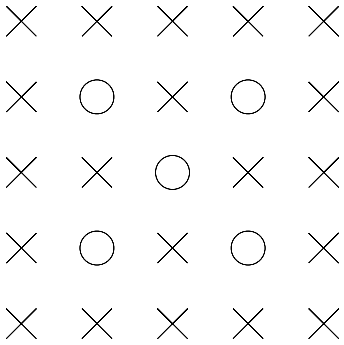
FIG. 21. h_+ (eq. 17) for the high resolution run of Fig. 19, compared to low and medium resolution runs with the boundaries twice as far out. The farther boundary wave amplitudes are extracted at $z = 8$ and scaled to account for the $1/r$ falloff. The structure differs markedly, especially around $t = 10 - 15$.

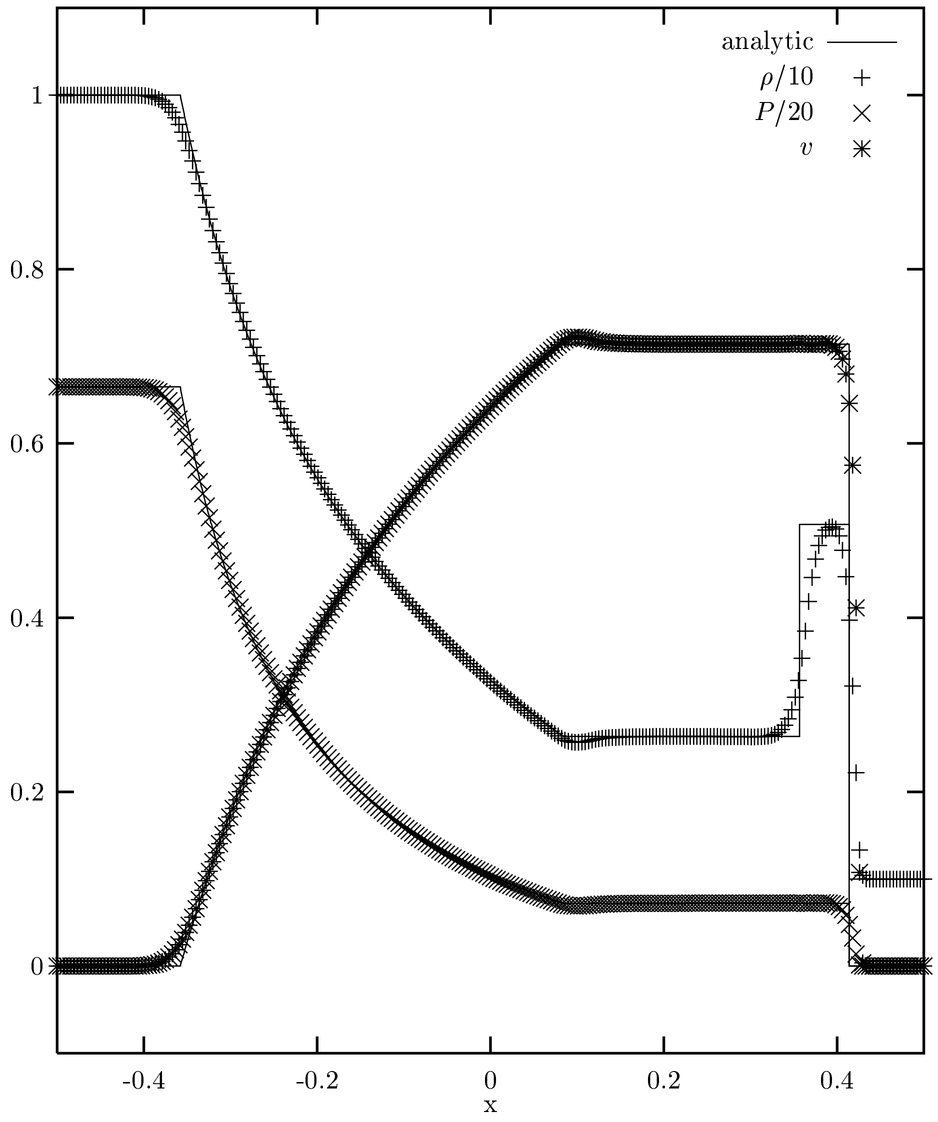
FIG. 22. $h_\times$ (eq. 18) for the high resolution run of Fig. 20, compared to low and medium resolution runs with the boundaries twice as far out. The farther boundary wave amplitudes are extracted at $z = 8$ and scaled to account for the $1/r$ falloff. The initial peak is significantly stronger and arrives later.

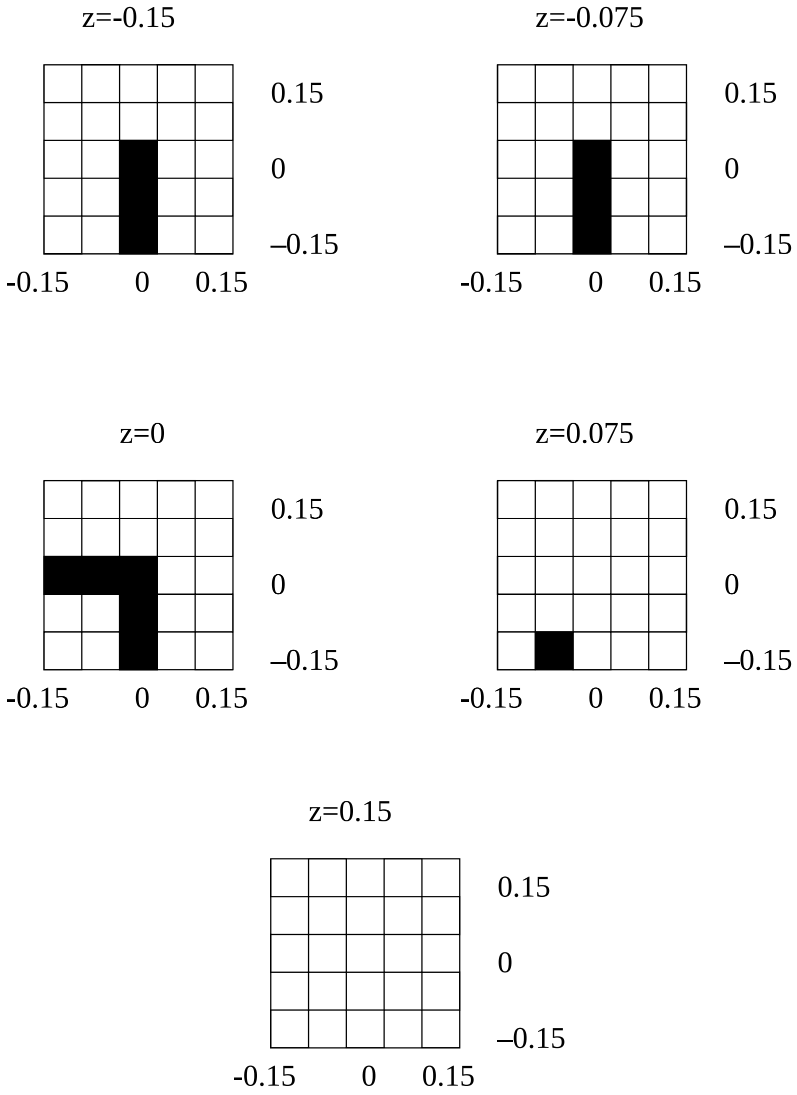
FIG. 23. h_+ for a low resolution run (65^3 points) with the boundaries at $(-8, 8)$. The amplitudes are extracted at $z = 4$ and $z = 8$. They do not match up, suggesting that the differences in Fig. 21 may be due to propagation effects.

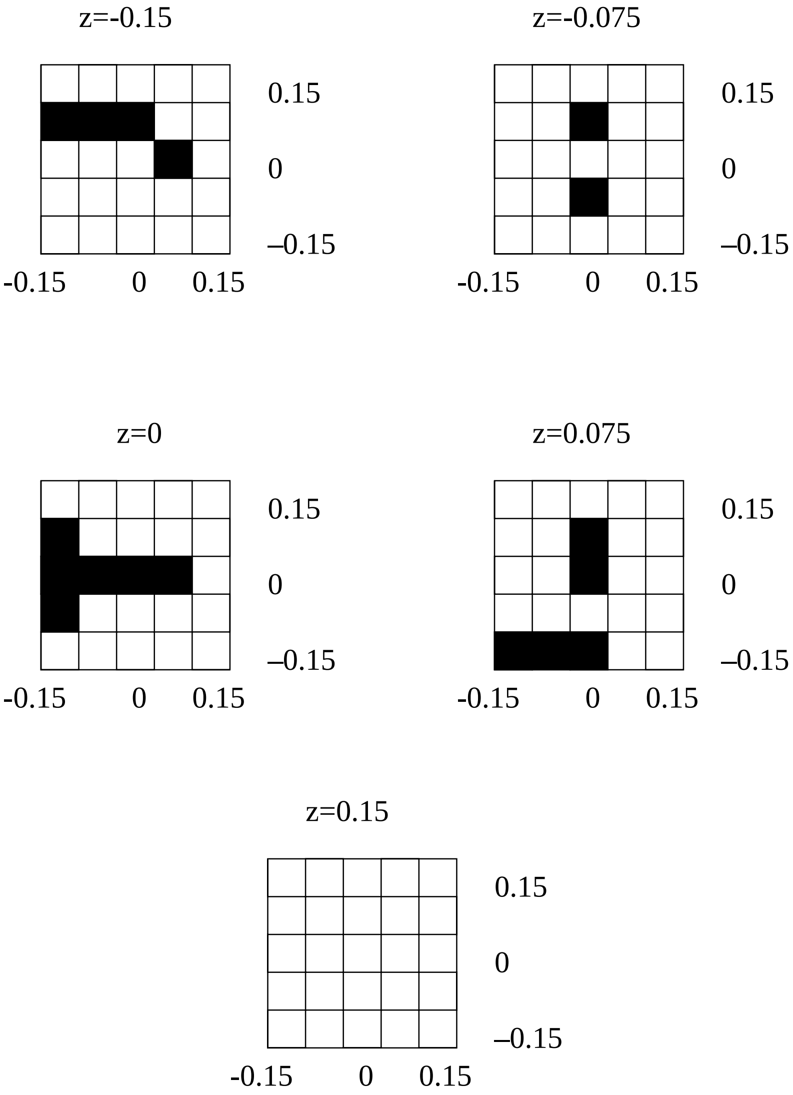
FIG. 24. $h_\times$ for a low resolution run (65^3 points) with the boundaries at $(-8, 8)$. The amplitudes are extracted at $z = 4$ and $z = 8$. They match up reasonably well, especially the large initial peak.






analytic
$\rho/10$
$P/20$
v
1
0.8
0.6
0.4
0.2
0
-0.4
-0.2
0
0.2
0.4
x


z=-0.15
0.15
0
–0.15
-0.15
0
0.15
z=-0.075
0.15
0
–0.15
-0.15
0
0.15
z=0
0.15
0
–0.15
-0.15
0
0.15
z=0.075
0.15
0
–0.15
-0.15
0
0.15
z=0.15
0.15
0
–0.15
-0.15
0
0.15



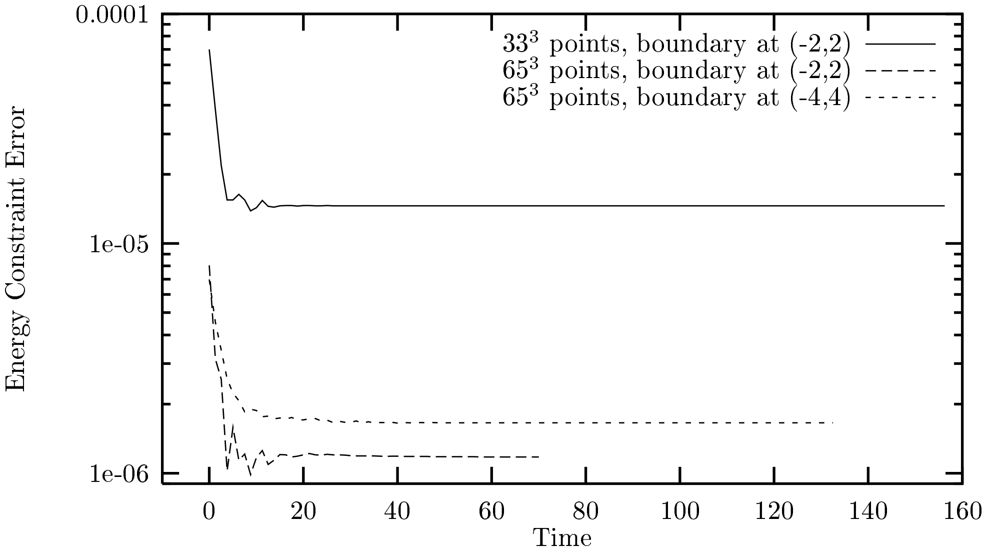
z=-0.15

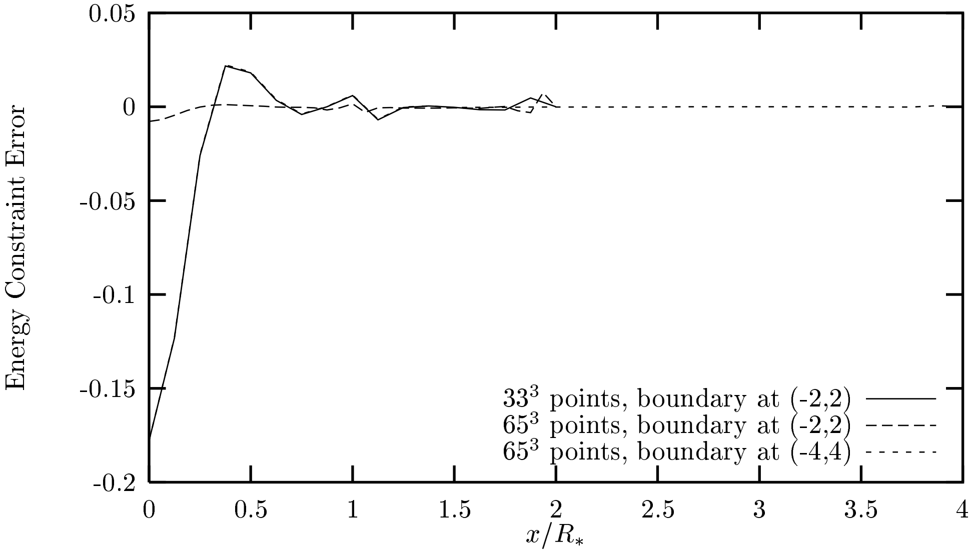
z=-0.075

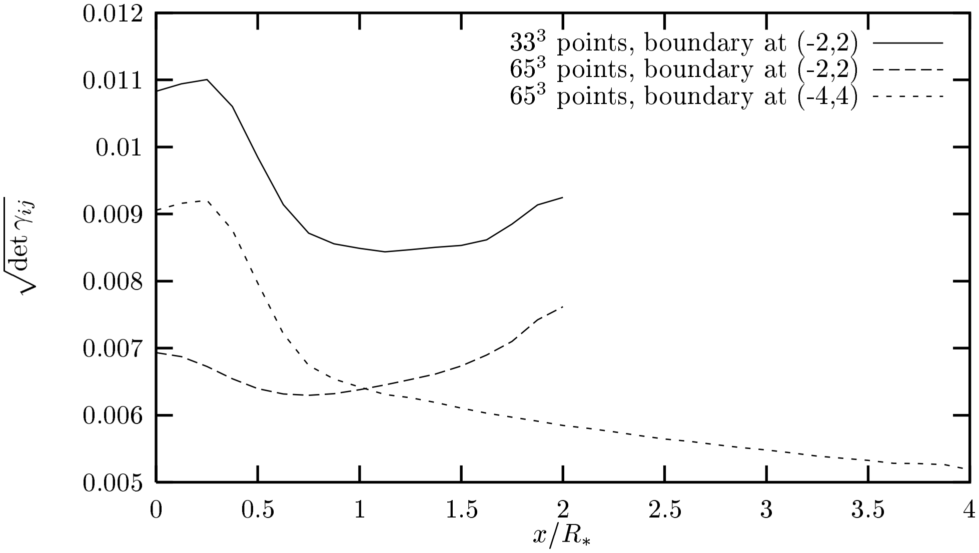
z=0

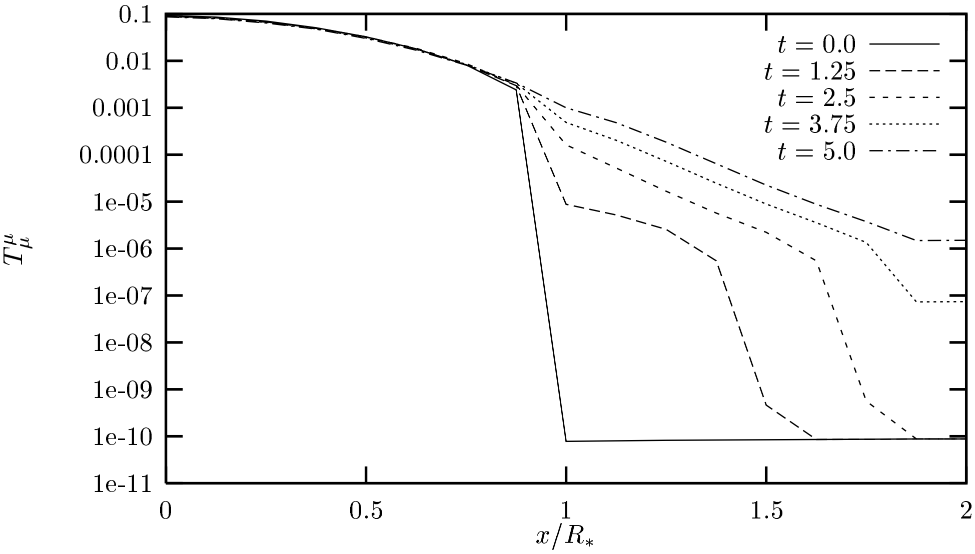
z=0.075

z=0.15

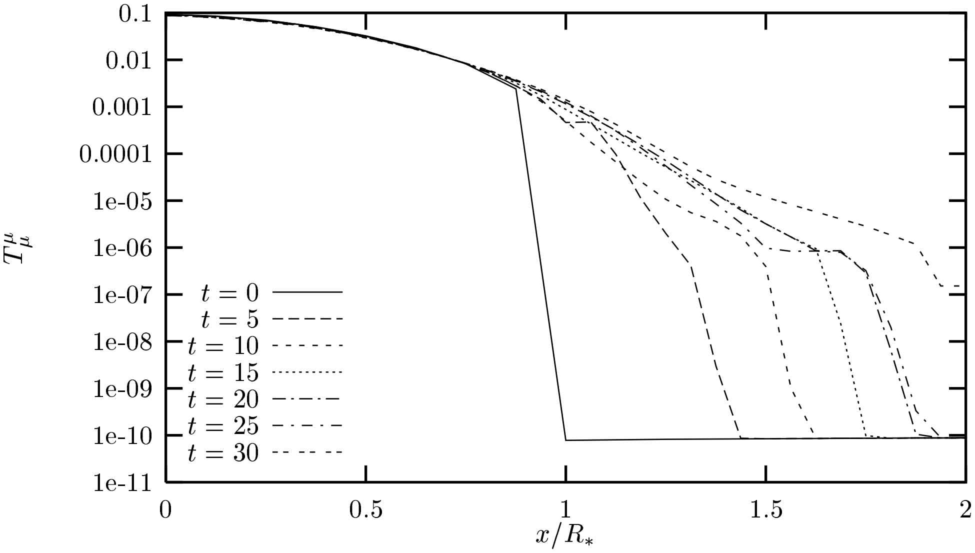

Energy Constraint Error
0.0001
1e-05
1e-06
-20
0
20
40
60
80
100
120
140
160
Time
33^3 points, boundary at (-2,2)
65^3 points, boundary at (-2,2)
65^3 points, boundary at (-4,4)

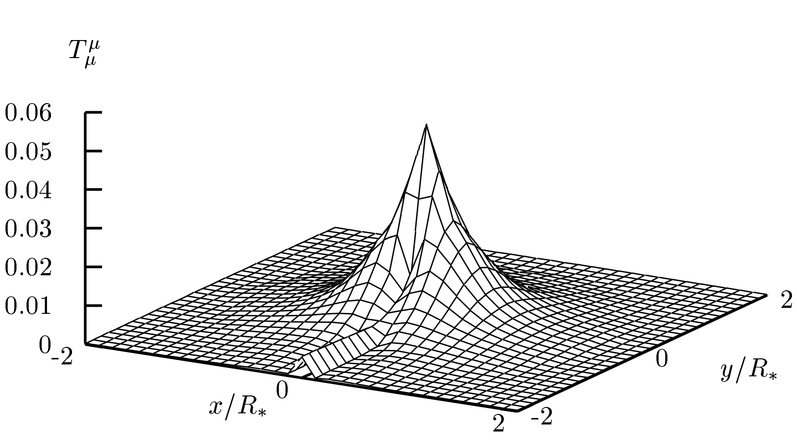

Energy Constraint Error
0.05
0
-0.05
-0.1
-0.15
-0.2
0
0.5
1
1.5
2
2.5
3
3.5
4
x/R_*
33^3 points, boundary at (-2,2)
65^3 points, boundary at (-2,2)
65^3 points, boundary at (-4,4)

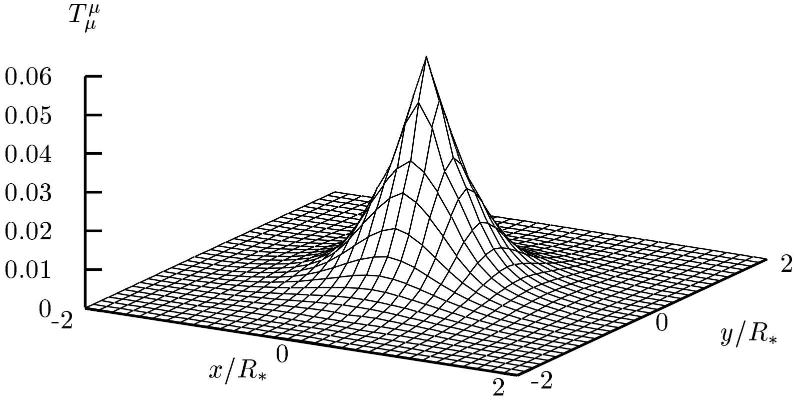

0.012
0.011
0.01
0.009
0.008
0.007
0.006
0.005
$\sqrt{\det \gamma_{ij}}$
0
0.5
1
1.5
2
2.5
3
3.5
4
x/R_*
33^3 points, boundary at (-2,2)
65^3 points, boundary at (-2,2)
65^3 points, boundary at (-4,4)

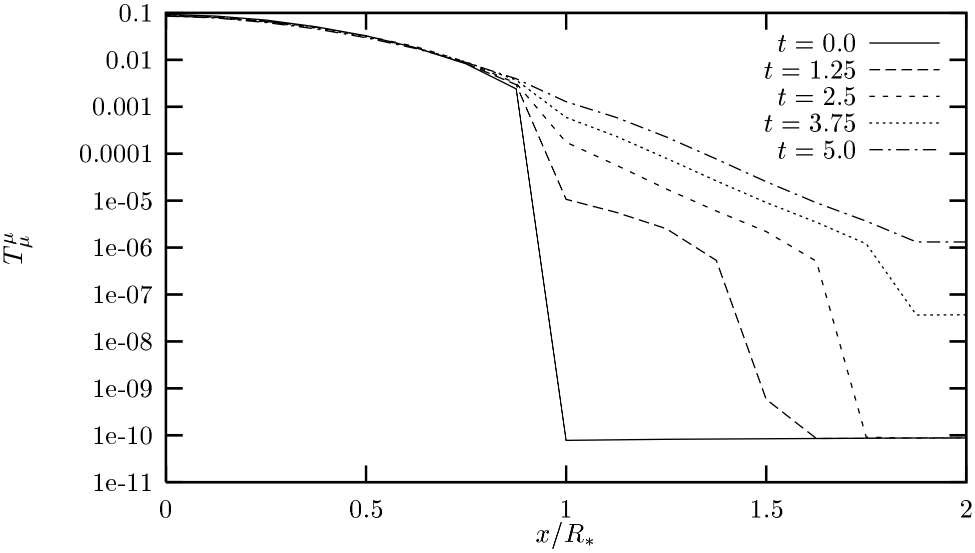


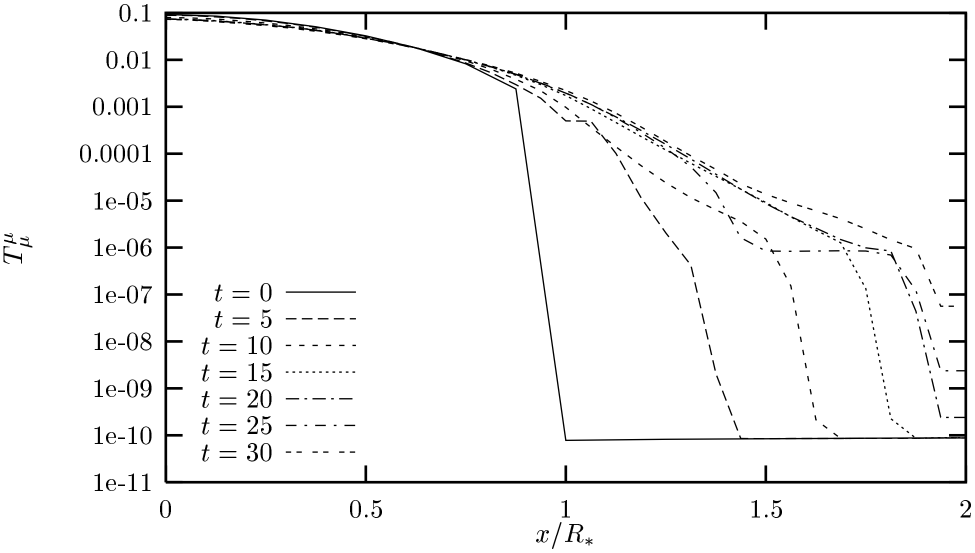
T^μ_μ
0.1
0.01
0.001
0.0001
1e-05
1e-06
1e-07
1e-08
1e-09
1e-10
1e-11
0
0.5
1
1.5
2
x/R_*
$t = 0.0$
$t = 1.25$
$t = 2.5$
$t = 3.75$
$t = 5.0$

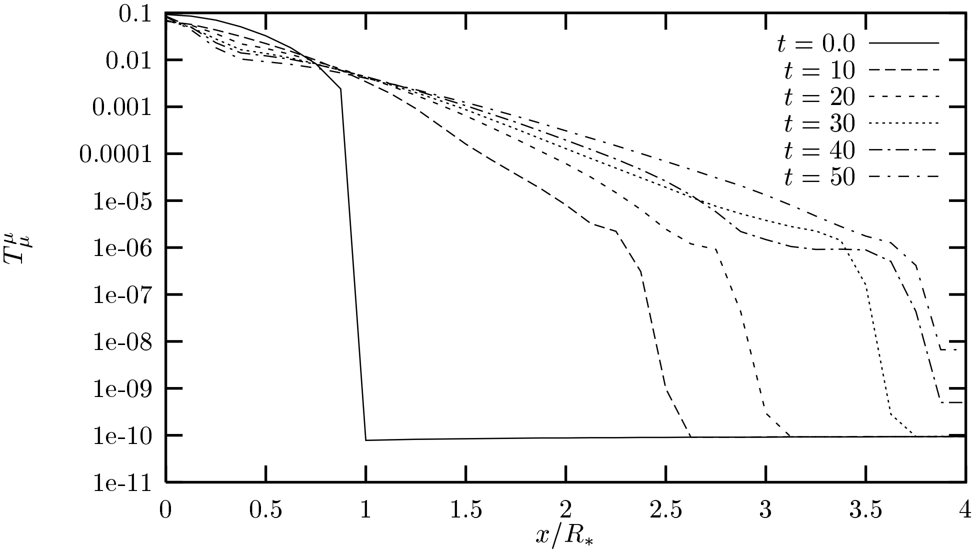

T^μ_μ
0.1
0.01
0.001
0.0001
1e-05
1e-06
1e-07
1e-08
1e-09
1e-10
1e-11
0
0.5
1
1.5
2
x/R_*
$t = 0$
$t = 5$
$t = 10$
$t = 15$
$t = 20$
$t = 25$
$t = 30$

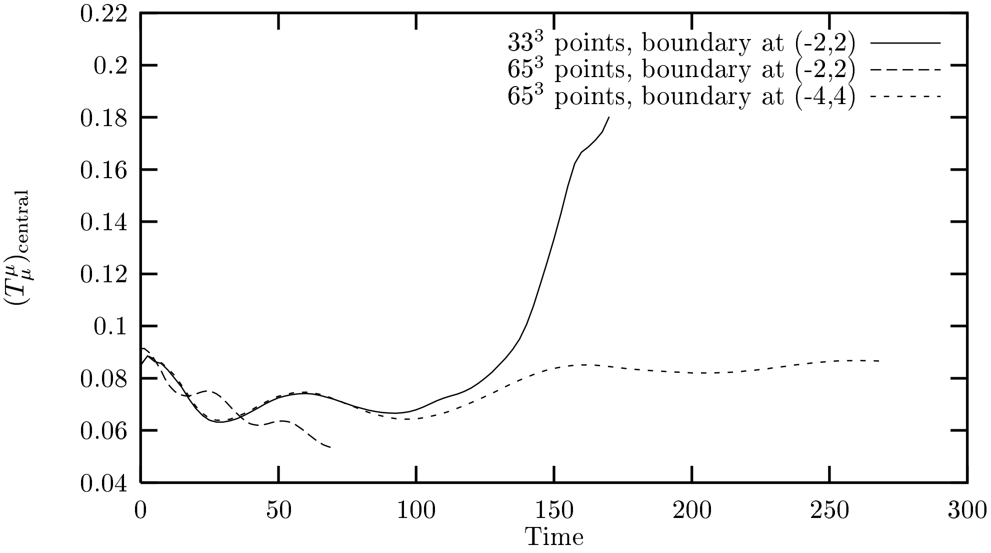

T_μ^μ
0.06
0.05
0.04
0.03
0.02
0.01
0
-2
0
2
x/R_*
-2
0
2
y/R_*


T^{μ}_{μ}
0.06
0.05
0.04
0.03
0.02
0.01
0
-2
0
x/R_*
2
-2
0
2
y/R_*

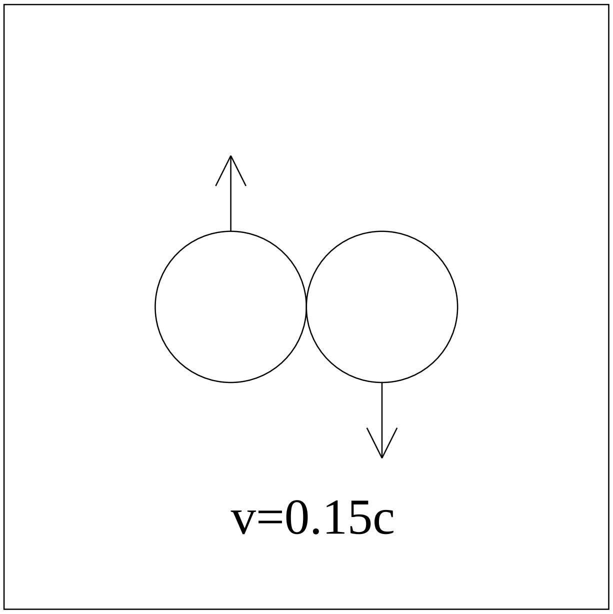

T^{μ}_{μ}
x/R_*
0.1
0.01
0.001
0.0001
1e-05
1e-06
1e-07
1e-08
1e-09
1e-10
1e-11
0
0.5
1
1.5
2
$t = 0.0$
$t = 1.25$
$t = 2.5$
$t = 3.75$
$t = 5.0$

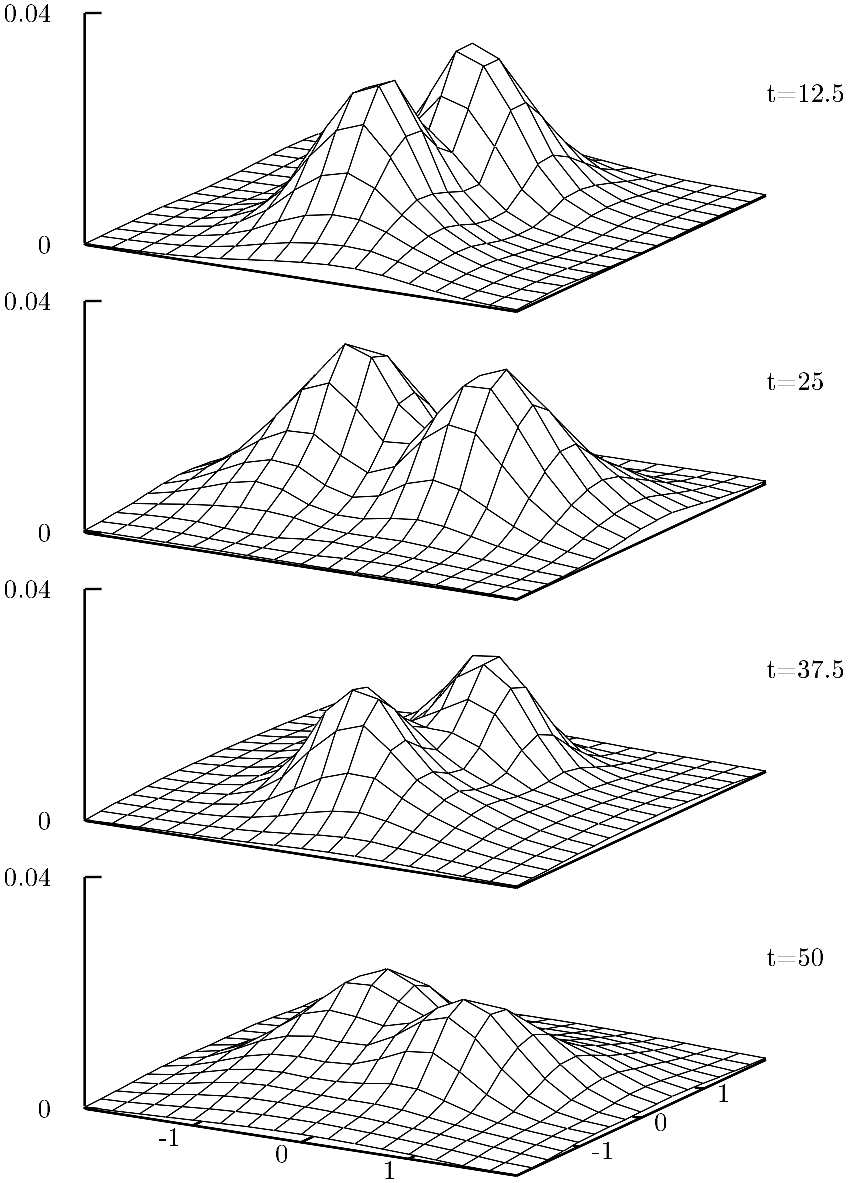

0.1
0.01
0.001
0.0001
1e-05
1e-06
1e-07
1e-08
1e-09
1e-10
1e-11
T^μ_μ
0
0.5
1
1.5
2
x/R_*
$t = 0$
$t = 5$
$t = 10$
$t = 15$
$t = 20$
$t = 25$
$t = 30$

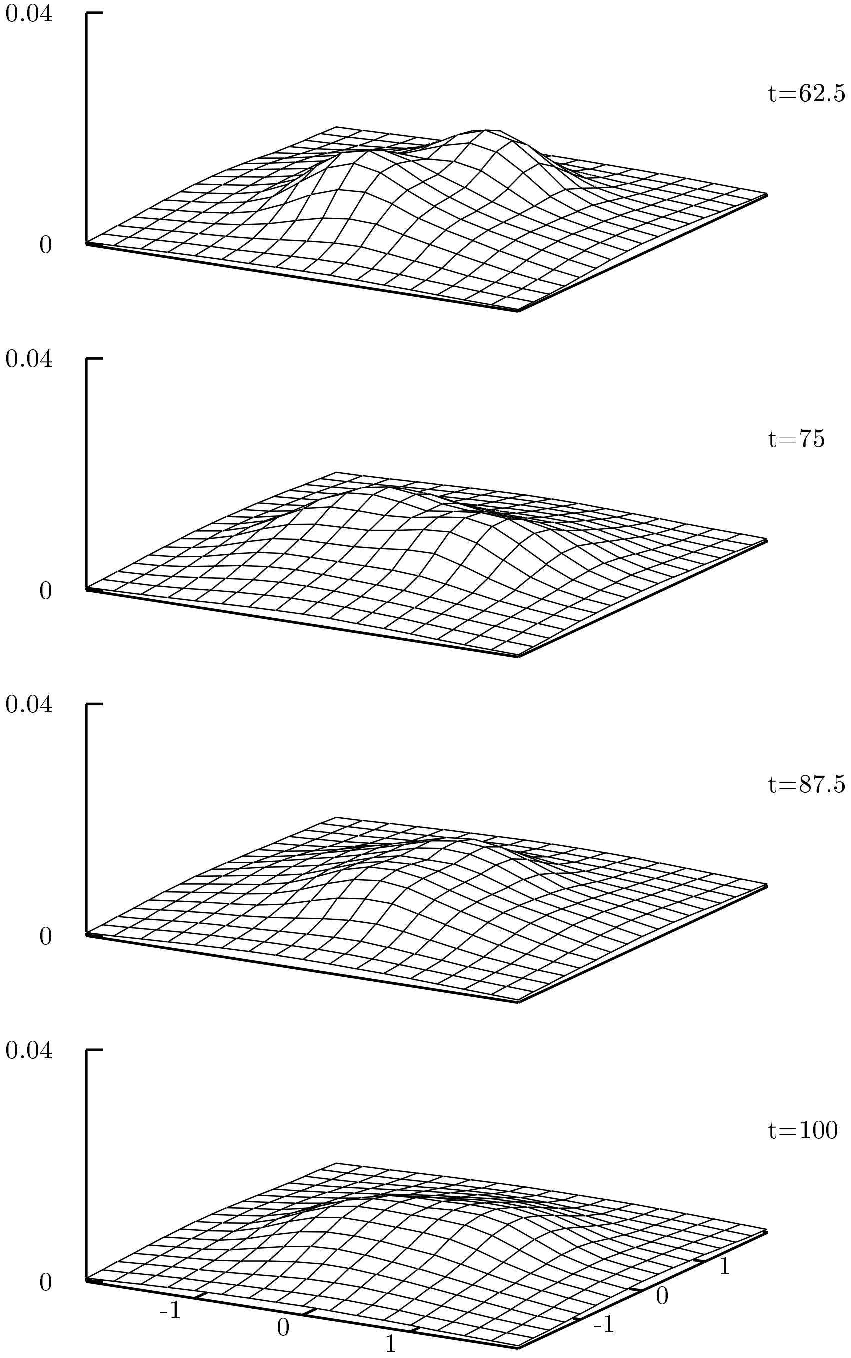

T^{μ}_{μ}
x/R_*
$t = 0.0$
$t = 10$
$t = 20$
$t = 30$
$t = 40$
$t = 50$
0.1
0.01
0.001
0.0001
1e-05
1e-06
1e-07
1e-08
1e-09
1e-10
1e-11
0
0.5
1
1.5
2
2.5
3
3.5
4

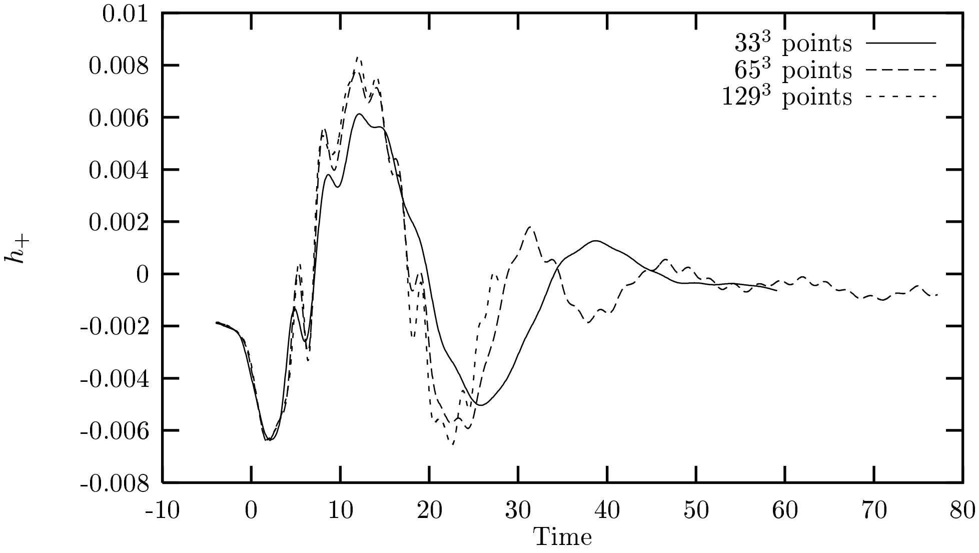


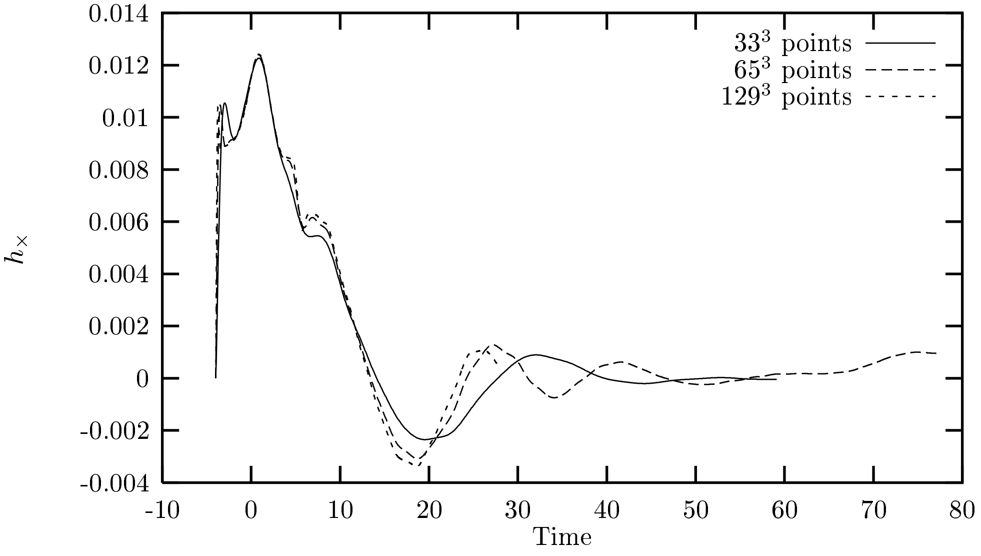
33^3 points, boundary at (-2,2)
65^3 points, boundary at (-2,2)
65^3 points, boundary at (-4,4)
$(T^\mu_\mu)_{\mathrm{central}}$
0.22
0.2
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0
50
100
150
200
250
300
Time


v=0.15c

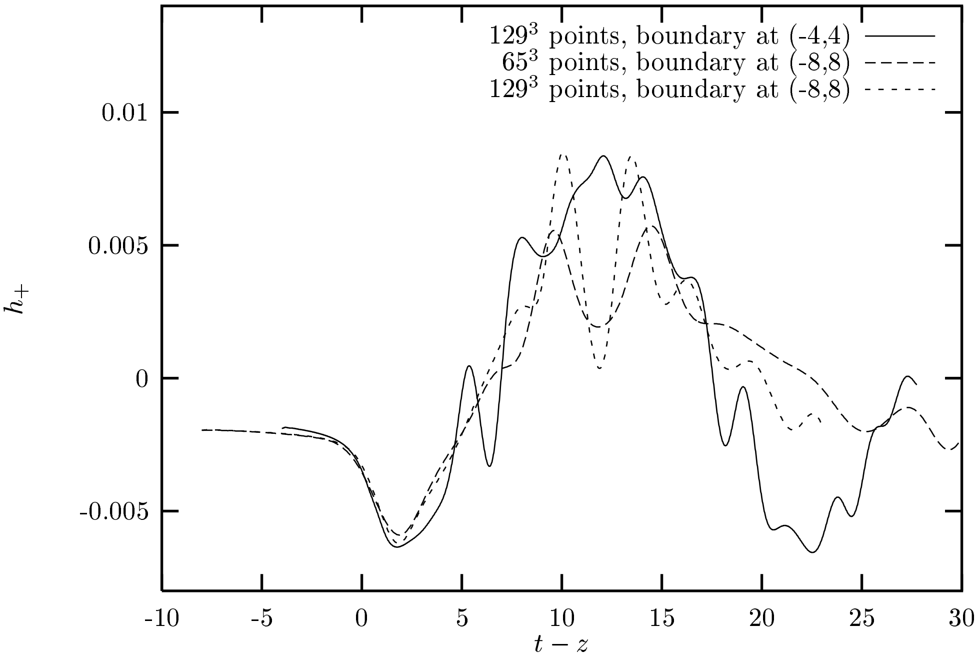

0.04
0
t=12.5
0.04
0
t=25
0.04
0
t=37.5
0.04
0
t=50
-1
0
1
-1
0
1


0.04
0
t=62.5
0.04
0
t=75
0.04
0
t=87.5
0.04
0
t=100
-1
0
1
-1
0
1

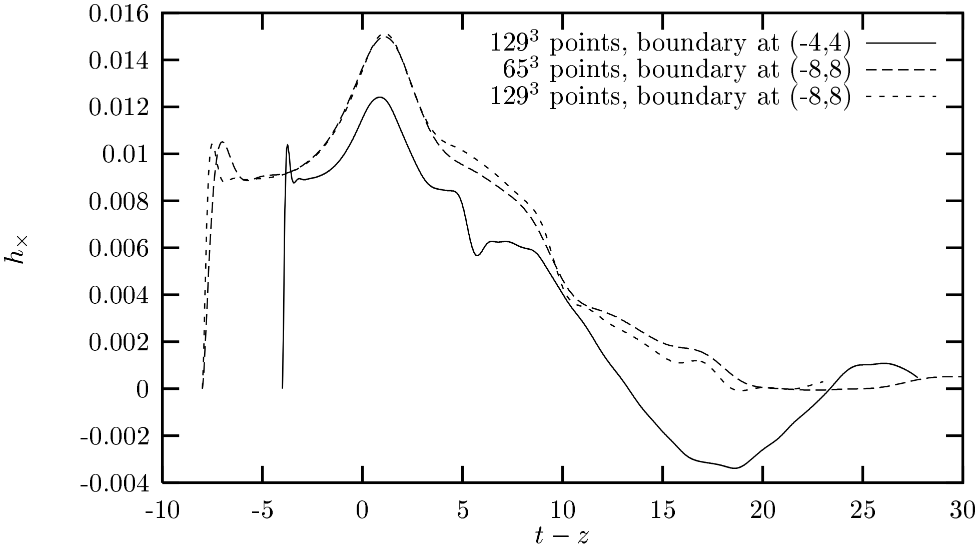

h_+
Time
0.01
0.008
0.006
0.004
0.002
0
-0.002
-0.004
-0.006
-0.008
-10
0
10
20
30
40
50
60
70
80
33^3 points
65^3 points
129^3 points

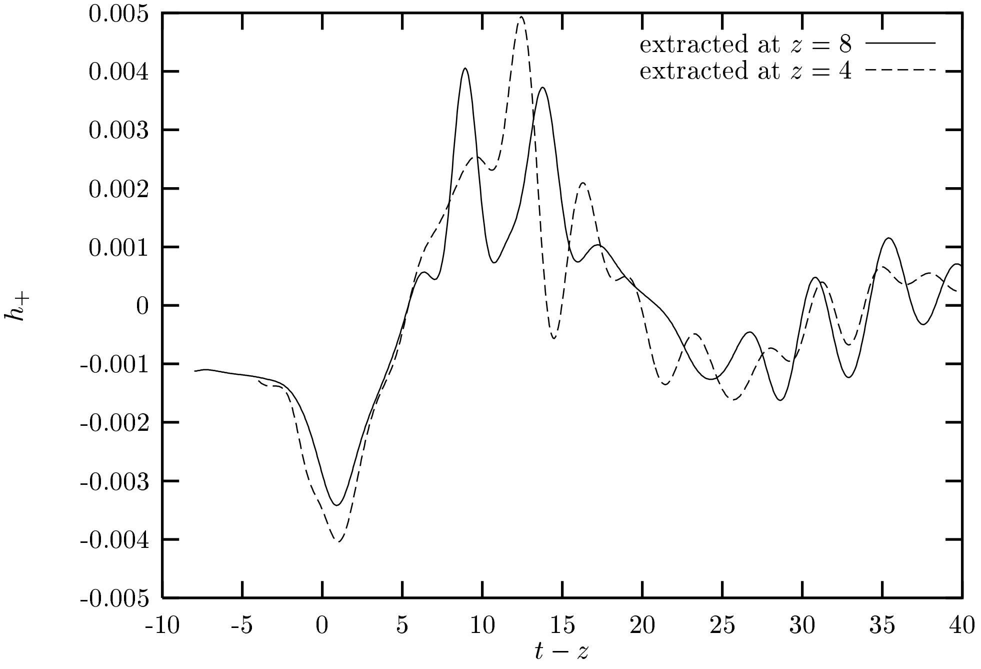


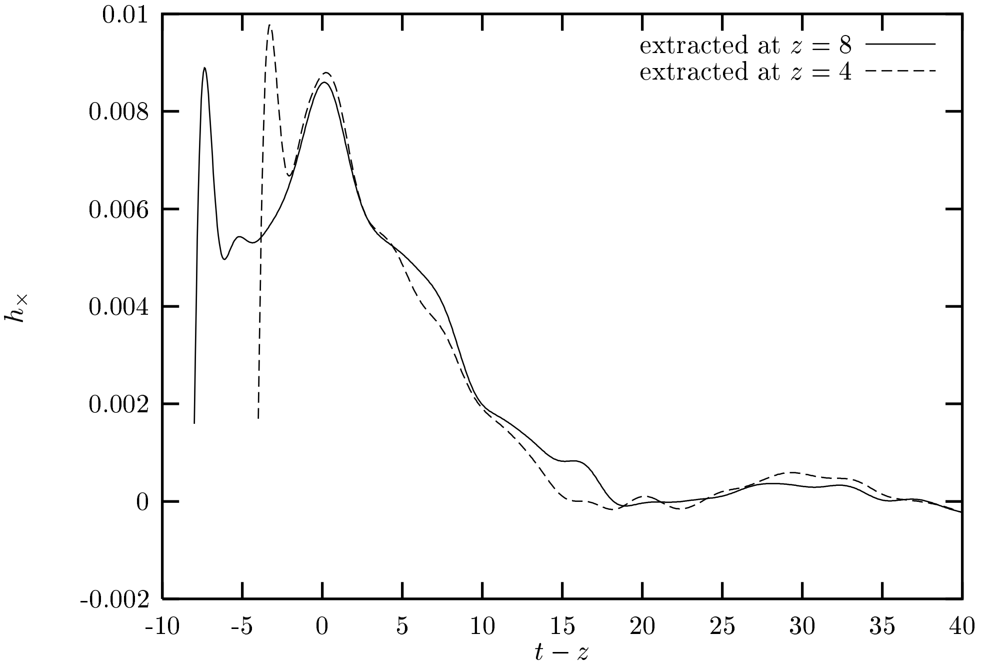
$h_\times$
0.014
0.012
0.01
0.008
0.006
0.004
0.002
0
-0.002
-0.004
-10
0
10
20
30
40
50
60
70
80
Time
33^3 points
65^3 points
129^3 points



129^3 points, boundary at (-4,4)
65^3 points, boundary at (-8,8)
129^3 points, boundary at (-8,8)
h_+
0.01
0.005
0
-0.005
-10
-5
0
5
10
15
20
25
30
$t - z$


129^3 points, boundary at (-4,4)
65^3 points, boundary at (-8,8)
129^3 points, boundary at (-8,8)
$h_\times$
0.016
0.014
0.012
0.01
0.008
0.006
0.004
0.002
0
-0.002
-0.004
-10
-5
0
5
10
15
20
25
30
$t-z$


extracted at $z = 8$
extracted at $z = 4$
h_+
$t - z$
0.005
0.004
0.003
0.002
0.001
0
-0.001
-0.002
-0.003
-0.004
-0.005
-10
-5
0
5
10
15
20
25
30
35
40



extracted at $z = 8$
extracted at $z = 4$
$h_\times$
$t - z$
0.01
0.008
0.006
0.004
0.002
0
-0.002
-10
-5
0
5
10
15
20
25
30
35
40